논문 99-24-11B-6 한국통신학회논문지 '99-11 Vol.24 No.11B

연결지향적 망 환경에서의 멀티미디어 서비스를 위한 동기화 기법

정회원 우 미 애*

A Synchronization Scheme for Multimedia Services in a Connection-Oriented Network Environment

Miae Woo* *Regular Member*

요 약

본 논문에서는 연결지향적 망 환경에서의 멀티미디어 데이터 동기 구현을 위하여 멀티미디어 서버에서의 전송 스케쥴링 기법을 제안하였다. 멀티미디어 서버와 클라이언트간에 대역폭과 지연 특성이 다른 다수의 채널이 존재한다는 가정 하에서, 멀티미디어 데이터 동기를 맞출 수 있는 전송 스케쥴링 문제를 병렬 프로세서의 스케쥴링 문제로 설정하였다. 문제가 NP-hard 문제이므로, 본 논문에서는 멀티미디어 문서를 구성하는 전송 데이터가 n개이고, 가용한 채널 수가 m일 때 시간적 복잡성이 $O(n \log n + nm)$인 두 개의 휴리스틱 알고리즘, FSA와 BSA를 제안하였다. 최적해를 구할 수 있는 알고리즘을 개발하여 시뮬레이션 한 결과 제안된 알고리즘들이 최적에 근사한 결과를 얻음이 확인하였고, 다양한 채널 시스템 조건이 제안한 두 알고리즘의 성능에 영향을 미침을 알 수 있었다.

ABSTRACT

In this paper, the problem of multimedia synchronization based on scheduling the transmission of multimedia documents in a connection-oriented network environment is investigated. Assuming channels with different bandwidth and delay characteristics are established between the multimedia server and the client, the scheduling problem is formulated to ensure stream synchronization as a parallel processor scheduling problem. Since the problem is NP-hard, we propose two heuristic algorithms, namely FSA and BSA, with time complexity $O(n \log n + nm)$, where n is the number of data units to be scheduled in the request multimedia document and m the number of channels available. Also an enumerative algorithm is developed to obtain the exact solutions. Extensive computational simulations reveal that the proposed consistently obtain near-optimal solutions. From the simulation results, we also identify special characteristics of the available channels which affect the relative performance of the proposed algorithms.

I. 서 론

분산 멀티미디어 시스템은 멀티미디어 데이터를 망을 통하여 전송하고, 주어진 시간적인 상관관계에 맞도록 동기를 유지하면서 재생하여야 한다. 그러므로 이러한 시스템에서는 망에서 발생하는 불특정 지연 때문에 야기될 수 있는 동기 유실을 복원하기 위하여 멀티미디어 서버와 클라이언트 양단에서 동기 복원을 위한 기법이 필요하다[1]. 실시간 데이터

* 세종대학교 정보통신공학과 (mawoo@sejong.ac.kr)
논문번호 : 99326-0817, 접수일자 : 1999년 8월 17일
※ 본 연구는 한국학술진흥재단 신진교수과제(1998-003-E00389) 지원으로 수행되었습니다.

및 저장형 데이터 응용 프로그램을 위하여 다양한 동기화 방식들이 여러 문헌에서 제안되었다. 예를 들면, 동기화 마커 방식을 사용한 실시간 데이터의 스큐 문제 해결을 위한 트랜스포트 프로토콜[2], 멀티캐스팅 환경에서 네트워크 지연을 이용한 클라이언트 단에서의 실시간 스트림 동기화[3], 피드백 정보를 사용한 분산 재생 시스템들 간의 동기 유실 극복 방안[4], 서버에서 멀티미디어 데이터의 전송 스케쥴링 방식[5] 등이 있다.

위에서 기술한 방식들은 응용프로그램에서 요구하는 QoS를 망에서 제공한다는 가정을 하였다. 그러나 패킷/셀 교환 망에서는 호 수락 제어의 결과로 일부의 호 설정 요구가 거부당할 수도 있다. 또한, 회선교환기를 사용하는 경우, 각각의 채널이 충분한 QoS를 제공하지 못하면 역 다중화를 통하여 여러 개의 채널을 이용하여 멀티미디어 데이터를 전송할 필요가 발생할 수도 있다. 이러한 경우 모두 망에서 응용프로그램이 원하는 QoS를 제공하지 못하는 사례에 해당한다고 할 수 있다.

본 논문에서는 망에서 응용프로그램이 요구하는 적절한 QoS를 수용하는 채널들을 제공하지 못하는 경우의 멀티미디어 데이터 동기화 문제를 논하고, 동기 복원을 위한 서버에서의 스케쥴링 방식을 제안한다. 고려하는 망에서는 채널이 연결지향적으로 설정되고, 호 설정 시 할당된 자원은 호가 종료될 때까지 가용하여 설정된 채널의 대역폭과 최대 지연에 대한 보장을 한다고 가정한다. 본 논문에서 고려하는 응용프로그램은 일정한 시간적 제약에 따라 기 편집, 저장된 멀티미디어 데이터를 사용한다고 가정하고 이를 "멀티미디어 문서"라 칭한다. 위에서 설정한 환경에서의 전체적인 시스템을 그림 1에서 보여준다.

본 논문에서는 연결지향적 망에서의 멀티미디어 데이터 동기 유지를 위한 전송 스케쥴링 문제를 deterministic, nonpreemptive 유니폼 병렬 프로세서 스케쥴링 문제로 공식화한다. 이 문제는 NP-hard 문제이고, 멀티미디어 서버는 클라이언트가 멀티미디어 문서를 접근할 때 실시간으로 스케쥴을 작성하여야 하므로, 서버에서 사용하는 알고리즘은 이러한 실시간 요구조건을 만족시킬 만큼 빠른 것이어야 한다. 이 문제를 해결하기 위하여, 시간적 복잡성이 $O(n \log n + nm)$인 휴리스틱 알고리즘, FSA (forward scheduling algorithm)과 BSA(backward scheduling algorithm)을 제안한다. 시간적 복잡성에서 n은 멀티미디어 문서를 구성하는 데이터 수를 의미하고, m은 연결지향적 망에서 가용한 채널 수

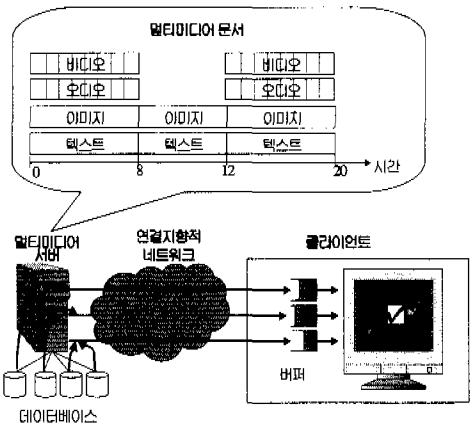


그림 1. 연결지향적 네트워크에서의 분산 멀티미디어 시스템 개념도

를 의미한다. 제안한 두 알고리즘들의 성능을 시뮬레이션을 통하여 평가한 결과, 제안한 알고리즘들은 적절한 시간 내에 최적 해에 근접한 결과를 지속적으로 얻음을 알 수 있다. 또한 다양한 채널 시스템 조건에서 제안한 두 알고리즘의 수행 결과로 얻은 멀티미디어 문서 재생 시작 시까지의 지연과 클라이언트에서 필요한 버퍼의 크기를 상호 비교하여, 조건에 따라 FSA와 BSA중 어느 알고리즘을 선택하여야 하는 지 제안한다.

본 논문의 구성은 다음과 같다. Ⅱ절에서는 서버에서의 스케쥴링 기법에 관련된 현안들이 제시된다. Ⅲ절에서는 이기종 채널 시스템에서의 멀티미디어 데이터 전송 스케쥴링 문제를 병렬 프로세스 스케쥴링 문제로 공식화한다. Ⅳ절에서는 FSA와 BSA를 제안하고, 최적해를 구하기 위한 알고리즘을 소개한다. 시뮬레이션을 이용한 실험을 Ⅴ절에 기술하고, Ⅵ절에서 결론을 맺는다.

Ⅱ. 멀티미디어 문서의 스케쥴링 상 현안들

멀티미디어 문서는 일반적으로 멀티미디어 문서 저자가 합성/저장 시 재생 스케쥴을 지정한다. 멀티미디어 문서의 데이터 단위를 본 논문에서는 "멀티미디어 객체"라 칭한다. 각각의 객체 저장 시 객체의 특성, 즉 그 객체의 종류, 크기, QoS 요구조건 등도 객체와 함께 저장된다고 가정한다.

멀티미디어 문서의 시간적 구조는 그림 2와 같이 표현할 수 있다. 그림 2 (a)에서 볼 수 있듯이, 전체 멀티미디어 재생을 여러 개의 시간 구간에 해당

하는 멀티미디어 객체를 동시에 재생하는 일련의 과정으로 볼 수 있다. 각각의 구간은 그 구간 안에 속하는 객체의 재생기간에 해당한다. 그러므로 멀티미디어 문서의 구조는 구간의 집합으로 기술될 수 있고, 각각의 구간은 지정된 기간과, 소속된 객체로 표시될 수 있다. 이러한 시간적 관계와 객체의 재생기간을 기준으로 하여 각 객체의 재생 마감시간을 유추할 수 있다.

그러나 멀티미디어 객체는 분산환경에서 통신과 동기화를 위한 단위로는 너무 단위가 크다. 멀티미디어 객체의 전송과 재생을 제어하기 위한 데이터 단위는 각각의 멀티미디어 객체를 보다 짧은 재생기간을 갖는 일련의 하부객체 (SIU : synchronization interval unit)로 나누어 얻을 수 있다[6].

일반적으로 비디오인 경우 SIU는 하나의 MPEG 프레임으로 설정할 수 있고, 한 프레임의 재생기간이 최소 동기화 단위가 된다. 이러한 논리에 따라 다른 종류의 실시간 멀티미디어 객체도 동기화 단위를 비디오 SIU의 동기화 단위에 맞추어 일련의 SIU로 분할할 수 있다. 그러나 비실시간 멀티미디어 객체, 즉 텍스트나 이미지는 객체 자체 내에 부가된 시간정보가 없기 때문에 하나의 객체를 하나의 SIU로 지정하면 된다. 그림 2 (a)에서 멀티미디어 객체로 표현한 멀티미디어 문서를 SIU를 사용하여 표현한 형식이 그림 2 (b)에 있다.

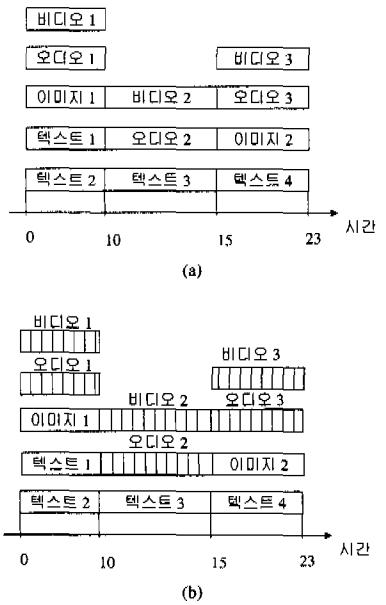


그림 2. 멀티미디어 문서의 구조 예 (a) 멀티미디어 객체 표현 (b) SIU 표현

이렇게 멀티미디어 객체를 일련의 SIU들로 분할하면, 멀티미디어 객체의 전송은 일련의 SIU들의 스트림 전송을 의미한다. 각 SIU는 소속 객체의 미디어 종류와 QoS 요구조건을 상속받는다. 또한 각각의 SIU에 대한 재생 마감시간은 그 SIU가 소속된 멀티미디어 객체의 재생 마감시간을 이용하여 구할 수 있다. 궁극적으로 멀티미디어 문서의 동기화는 SIU들을 각각의 재생 마감시간 전에 클라이언트에게 전달해 줌으로써 이룰 수 있다.

Ⅲ. 스케줄링 문제

만일 연결지향적 망에서 요구되는 QoS 인자를 지원할 수 있는 일련의 채널들을 제공하지 못하는 경우, 클라이언트의 요청을 거부하는 대신, 그림 1에서 보여주듯이 일정량의 멀티미디어 데이터를 클라이언트 단에 미리 인출하여 저장하였다가 추후에 재생할 수도 있다. 이러한 방식은 클라이언트 단에서 멀티미디어 객체를 재생하기 전에 버퍼링 하기 위한 초기 지연을 야기하므로 초기 지연을 최소화하는 것이 최적화의 목표가 된다.

클라이언트가 검색하고자 하는 멀티미디어 문서가 전체적으로 n개의 SIU, $S=\{SIU_1, SIU_2, \ldots, SIU_n\}$,로 구성되었다고 가정한다. 각각의 SIU는 스케줄링과 관련하여 두 가지 인자, 즉 SIU_i의 크기 s_i와 재생 마감시간 d_i을 가지고 있다. 이기종 채널 시스템에서 m개의 채널들로 구성된 채널집합 $C=\{C_1, C_2, \ldots, C_m\}$이 가용하다고 가정한다. 각각의 채널 C_j은 실효 대역폭 c_j와 통과지연에 대한 상한값 δ_j를 보장한다.

스케줄링 방침에 의해 SIU_i가 채널 C_j에 시각 a_j에 전송하도록 스케줄이 정해졌다고 가정하면 SIU_i의 클라이언트 단 도착시간 A_i는

$$A_i = a_j + \frac{s_i}{c_j} + \delta_j$$

이 된다. 이러한 경우 재생 마감시간에 대한 SIU_i의 지연은 $T_i = \max\{0, A_i - d_i\}$로 정의된다. 만일 $T_i > 0$인 경우, SIU_i는 주어진 재생 마감시간을 맞추지 못하게 되고, 결과적으로 비동기를 유발하게 된다. SIU들이 각각의 재생 마감시간을 놓치지 않기 위해서는, 가장 늦게 클라이언트 단

에 도착하는 SIU들이 가용할 때까지 재생의 시작 시점을 늦추어야 한다. 즉, 각각의 SIU의 재생은 최대 지연 $T_{max} = \max_{1 \le i \le n}\{T_i\}$만큼 지연되어야 한다. 이러한 내용이 다음 명제에 기술되어 있다.

명제 1. $\min\{d_i\}=0$인 조건이 주어졌을 때, 멀티미디어 문서의 가장 빠른 가능한 재생 시작 시각은 최대지연을 도입한 $T_{max} = \max_{1 \le i \le n}\{T_i\}$이다.■

명제 1에서 알 수 있듯이 유도된 재생 마감시간 $(d_i + T_{max})$, $1 \le i \le n$,은 주어진 스케줄에 따라 동기화된 재생을 하면서 맞출 수 있는 가장 빠른 재생 마감시간이다. 원래의 재생 마감시간 d_i에 비하여 유도된 재생 마감시간의 T_{max}는 재생 과정의 초기 지연이다.

스케줄링 관점에서 볼 때, 초기 재생 지연을 최소화 할 수 있도록 주어진 m 채널에 n SIU들을 어떠한 순서로 전송할 지를 정하는 것이 현안으로 등장한다. 스케줄링 문제는 아래와 같이 기술할 수 있다.

문제: n개의 SIU의 집합 $S=\{SIU_1, SIU_2, \ldots, SIU_n\}$, 각각의 SIU의 크기와 재생 마감시간, m개의 채널들로 구성된 채널집합 $C=\{C_1, C_2, \ldots, C_m\}$과 각 채널의 대역폭과 통과 지연이 주어진 경우, 최소 T_{max}를 제공하는 m 채널상의 n SIU에 대한 스케줄을 구하라. ■

대역폭이 다른 m개의 채널들은 처리 속도가 다른 m개의 유니폼 프로세서들로 나타낼 수 있고, SIU의 집합은 독립적인 작업과 동등하게 볼 수 있다. 작업의 처리시간은 SIU의 크기로 주어진다. 멀티미디어 서버가 언제든지 원하는 멀티미디어 데이터를 접근할 수 있기 때문에 동시에 가용하다고 전제할 수 있다. SIU의 재생 마감시간은 정해진 수치이므로 스케줄링은 결정적 문제이다. 또한 SIU는 전송이 끝날 때까지 해당 채널을 점유하므로, 이것은 nonpreemptive 환경에 해당한다.

주어진 문제가 다른 프로세서 스케줄링 문제와 구별되는 점은 각 채널의 통과 지연 δ_j에 있다. 여기서 주목할 점은 통과 지연이 SIU와는 무관하다는 것이다. 전통적인 유니폼 프로세서 스케줄링 모델에서는 작업 i를 프로세서 j에서 처리하는 시간 p_{ij}이 s_i / c_j 라고 가정한다. 이것은 본 논문의 모델에서 모든 통과 지연이 0인 경우에 해당하는 특별한 경우이다.

본 논문에서 설정한 문제의 특별한 경우가 $\delta_j=0$, $c_j=c$ 인 경우이다. 이 경우는 m개의 동일한 병렬 프로세서에서 n개의 작업 스케줄 시 최대 지연을 최소화하는 스케줄링 문제와 동등하고, 이 병렬 프로세서 스케줄링 문제는 잘 알려진 NP-hard 문제이다[7]. 결과적으로 본 논문에서 다루는 문제도 NP-hard 문제이다.

Ⅳ. 스케줄링 알고리즘

본 절에서는 본 논문에서 풀고자 하는 스케줄링 문제에 대한 두 가지 휴리스틱 알고리즘을 먼저 소개하고, 그 후에 최적해를 구하는 알고리즘을 소개한다. 이러한 알고리즘의 소개에 앞서, 모든 알고리즘에서 유용하게 사용할 수 있는 최적해의 구조에 대한 통찰력을 제공하는 정리를 먼저 소개한다.

정리 1. 설정된 문제에 대하여 각 채널에 EDD(earliest due date) 순서로 스케줄이 정해져 있는 최적 스케줄이 존재한다. ■

정리 1의 결과로, 계산상 고려해야 하는 수색영역을 상당히 줄일 수가 있다. 또한 아래에서 제안하는 효과적인 휴리스틱 알고리즘의 기본이 된다.

1. 휴리스틱 스케줄링 알고리즘

지금부터 문제를 풀기 위하여 두개의 휴리스틱 알고리즘, FSA와 BSA를 제안한다.

이 두 알고리즘에서 집합 S는 정렬된 SIU들의 리스트를 지칭하고, $SIU_{(i)}$는 정렬된 리스트에서 i번째 SIU를 나타낸다. a_j는 채널 부하가 없는 가용시간을 나타낸다. a_j는 SIU가 채널 C_j에 할당될 때마다 갱신된다.

FSA에서는 초기 재생에 필요한 SIU부터 시작하여 재생 마감시간이 증가하는 순서대로 스케줄을 정한다. FSA 알고리즘을 기술하면 다음과 같다.

FSA1. S 안의 각 SIU를 재생 마감시간이 증가하는 순서로 정렬한다. 만일 두 SIU가 동일한 재생 마감시간을 가지고 있으면, 큰 크기의 SIU가 우선한다.

FSA2. $L_j=\emptyset$, $a_j=0\ \forall j$ 로 초기화한다.

FSA3. 리스트의 앞에서부터, S 안의 각 SIU를 클라이언트 측에서의 도착시간을 최소화하는

채널로 스케줄링 한다. 즉, 만일 $j = \arg\min_{1\le k\le m}\left\{ a_k + \frac{s_{(i)}}{c_k} + \delta_k \right\}$ 이면

$L_j = L_j \cup \{ SIU_i \}$.

$A_{(i)} = a_j + \frac{s_{(i)}}{c_j} + \delta_j$.

FSA4. $a_j = a_j + \frac{s_{(i)}}{c_j}$

FSA5. $T^{\star}_{\max} = \max_{1\le i\le n}\{0, A_{(i)} - d_{(i)}\}$

BSA에서는 FSA와는 달리 제일 나중에 재생 할 SIU부터 시작하여 재생 마감시간이 감소하는 순서대로 스케줄을 정한다. BSA 알고리즘을 기술하면 다음과 같다.

BSA1. S 안의 각 SIU를 재생 마감시간이 증가하는 순서로 정렬한다. 만일 두 SIU가 동일한 재생 마감시간을 가지고 있으면, 작은 크기의 SIU가 우선한다.

BSA2. $L_j = \emptyset$, $a_j = \max_{1\le i\le n}\{d_i\} \forall j$ 로 초기화 한다.

BSA3. 리스트의 끝에서부터, S안의 각 SIU를 전송 시작 시간을 극대화하는 채널로 스케줄링 한다. 즉, 만일

$$j = \arg\max_{1\le k\le m}\{\min\{d_{(i)}, a_k\} - \frac{s_{(i)}}{c_k} - \delta_k\}$$

이면 $L_j = \{ SIU_i \} \cup L_j$. $A_{(i)} = \min\{d_{(i)}, a_k\}$.

BSA4. $a_j = A_i - \frac{s_{(i)}}{c_j}$

BSA5. $T^{\star}_{\max}$ 는

$$\max_{1\le i\le n}\{ A_{(i)} - d_{(i)} - \min_{1\le j\le m}\{a_j - \delta_j\}\}$$

위에서 기술한 FSA와 BSA는 모두 다 greedy 알고리즘으로써 정리 1에서 제시한 최적성의 조건을 만족시킨다. 두 알고리즘은 n개의 SIU를 정렬하는 데 $O(n \log n)$, SIU들을 m개의 채널에 스케줄링하는 데 $O(nm)$의 복잡성을 갖으므로 결과적으로 $O(n \log n + nm)$의 시간적 복잡성을 갖는다.

2. 최적해를 구하기 위한 알고리즘

이 절에서는 최적해를 구하기 위한 알고리즘으로 branch and bound 알고리즘을 제시한다. 본 논문에서 고려하고 있는 문제가 NP-hard이기 때문에, 고려해야하는 스케줄의 수는 SIU의 수나 채널 수가 증가함에 따라 지수적으로 증가하므로 주어진 문제에 대한 해를 다항시간 안에 구할 수 없다. 그러므로 branch and bound 수순을 잘 설계하여 실제로 평가하는 스케줄 수를 적게 유지하도록 하는 것이 중요하다.

Branch and bound 수순은 탐색 트리를 생성하는데, 이 트리의 k 레벨에서의 노드는 부분해를 나타낸다. 부분해는 시간상 증가하도록 구성된다. 부분해 $P(k)$는 k쌍의 (i_r, j_r), $1 \le r \le k$,로 구성이 되는 데, 여기서 i_r는 SIU의 인덱스를 나타내고 j_r는 SIU_{i_r}이 할당된 채널을 나타낸다. 탐색 트리는 k 레벨의 선택된 노드로 분기함으로써 확장한다. 현재의 부분해에 포함되어 있지 않은 SIU, $SIU_i \notin P(k)$,에 대하여 처음 k개의 SIU-채널 쌍은 $P(k)$에 있는 것으로, $(k+1)$번째 쌍은 (i_{k+1}, j_{k+1})이 되도록 트리에 새로운 노드를 추가함으로써 트리를 확장한다.

정리 1에 의하여, 만일 추가를 고려하는 쌍 (i_{k+1}, j_{k+1})이 $P(k)$ 안에 있는 쌍 (i_r, j_r)과 비교하여 $d_{i_{k+1}} < d_{i_r}$이고 $j_{k+1} = j_r$, $1 \le r \le k$,라면, 이 쌍으로 분기할 필요가 없다. 또한 $r = \arg\min_{1\le s\le k}\{ i_s > i_{k+1}\}$이고 $j_r \ne j_{k+1}$인 경우도 분기할 필요가 없다. 예를 들어 부분해 [(1,1), (2,2), (4,3), (5,3), (3,2)]와 [(1,1), (2,2), (3,2), (4,3), (5,3)]는 대등하다. 이렇게 함으로써 각 부분해 $P(k)$를 단 한번만 평가하고 탐색 과정에서의 불필요한 중복 과정을 제거할 수 있다.

Branch and bound 과정에서 탐색 전략으로 depth first search를 사용하여 상위한도(UB)를 갱신한다. 전 절에서 기술한 FSA에서 구한 $T^{\star}_{\max}$ 값을 UB의 초기값으로 선택한다. UB는 보다 작은 값의 $T_{\max}$를 갖는 스케줄이 구해지면 갱신한다. 부분해 $P(k)$는 하위한도(LB)와도 연관되어 있다. 만일 노드의 LB가 현재의 UB보다 더 값이 크거나 같다면 해당노드는 제거된다.

부분해의 LB는 preemption이 허용 안 되는 조건을 완화함으로써 계산한다. 이 LB를 구하기 위하여 유니폼 프로세서 시스템에서의 preemptive 스케줄링을 위해 제안된 수순[7]을 통과 지연 δ_j를 고려할 수 있도록 수정하여 사용한다. 이 과정은 아래와 같이 설명할 수 있다.

g_j를 구간 $[0, d]$에서 채널 C_j의 처리용량이라고 하자. 만일 채널 C_j가 a_j에 가용하다면

$g_j=(d-\alpha_j-\delta_j)\cdot c_j$가 된다. 이 과정에서 m개의 채널들이 각각의 처리용량에 따라 $g_j \geq g_{j+1}$가 되도록 정렬되었다고 가정한다. 재생 마감시간이 d로 동일한 SIU들이 n'개가 있다면, 이 SIU들은 크기 순으로, 즉 $s_i \geq s_{i+1}$, $1 \leq i < n'$,가 되도록 정렬되어 있다고 가정한다.

$n' \geq m$인 경우 $S_j=\sum_{i=1}^{j} s_i$, $1 \leq j < m$,와 $S_m=\sum_{i=1}^{n'} s_i$로 정의하자. 그러면 n'개의 SIU들은 만일 $\max_{1 \leq j \leq n'}\{S_j/\sum_{l=1}^{j} g_l\} \leq 1$라면 목적지에 재생 마감시간인 d 시간까지 preemption되면서 전달될 수 있도록 스케줄링이 가능하다. $n' < m$인 경우에는 $\max_{1 \leq j \leq n'}\{S_j/\sum_{l=1}^{j} g_l\} \leq 1$인 경우에만 재생 마감시간에 맞출 수 있도록 스케줄을 지정할 수 있다[8]. 이 내용은 branch의 유효성을 시험하는 데 사용된다.

d_l^*, $1 \leq l \leq L$,를 부분해 $P(k)$에 포함되어 있지 않은 $(n-k)$ 개의 SIU들의 재생 마감시간이라고 하자. 단 각각의 d_l^*는 서로 다른 값을 갖는다. d_l^*는 오름차순으로 정렬되었다고 가정한다. 즉, $d_l^* < d_{l+1}^*$을 만족시킨다. 그러면 L단계에서 구성하는 preemptive 스케줄은 다음과 같이 생성된다. $L_j(P(k))$를 부분해 $P(k)$ 중에서 채널 C_j에 해당된 부분 스케줄을 나타낸다고 하자. 부분 스케줄 $L_j(P(k))$에 의거하여 채널 C_j의 가용시간 α_j는 다음과 같이 설정한다.

$$\alpha_j=\frac{\sum_{SIU_i \in L_j(P(k))} s_i}{c_j}$$

g_j^l를 단계 l에서의 채널 C_j의 처리용량이라고 하고, R_j^l를 단계 l에서 SIU들을 할당한 뒤 남은 채널 C_j의 나머지 용량이라고 하자. 그러면 단계 l에서의 채널 C_j의 처리용량은 다음과 같이 된다.

$$g_j^l=\begin{cases} R_j^{l-1}+(d_l^*-d_{l-1}^*)\cdot c_j, & g_j^{l-1}>0 \text{ 인 경우} \\ \max\{(d_l^*+UB-\alpha_j-\delta_j)\cdot c_j, 0\}, & g_j^{l-1}=0 \text{ 이고} \\ \max\{d_i \mid SIU_i \in L_j(P(k))\} \leq d_l^* & \text{인 경우} \\ 0, & \text{그 외의 경우.} \end{cases} \quad (1)$$

식 (1)의 초기조건은 $d_0^*=g_j^0=R_j^0=0$이다. 위 식에서 g_j^l는 $L_j(P(k))$ 안의 SIU들 중에서 최대 재생 마감시간이 d_l^* 보다 크면 0으로 지정한다. 단계 l에서 $SIU_i \notin P(k)$이고 $d_i=d_l^*$ 인 SIU_i들의 스케줄 가능성을 시험한다[8].

Branch and bound 알고리즘을 정리하면 아래와 같다.

BB1. UB를 FSA의 $T_{\max}^*$로 초기화 한다.

BB2. 가능한 모든 스케줄을 고려할 때 까지 depth first search를 사용하여 부분해 $P(k)$를 구성하고, 만일 가능한 모든 스케줄을 다 고려한 경우에는 종료한다.

BB3. $k=n$인 경우, $T_{\max} < UB$이면 $UB=T_{\max}$로 갱신하고 현재의 $P(n)$을 최적해로, $T_{\max}^*=T_{\max}$로 기록한다. BB2로 간다.

BB4. $SIU_i \notin P(k)$의 d_i를 d_i+UB로 하여 스케줄 가능성을 시험한다. 만일 가능하면 $k=k+1$로 하고 BB2로 간다. 만일 불가능하면 해당노드를 제거하고 BB2로 간다.

V. 시뮬레이션을 통한 성능 분석 결과

제안한 FSA와 BSA의 성능을 분석하기 위하여 다양한 멀티미디어 문서 구조와 서로 다른 특성을 갖는 채널 시스템에 대하여 시뮬레이션을 하였다. 시뮬레이션은 최적해와의 성능 비교를 위한 시뮬레이션과 제안한 알고리즘들간의 상호 성능 비교를 목적으로 하는 시뮬레이션으로 구분하여 수행하였다.

II절에서 잠시 거론하였듯이, 멀티미디어 문서는 구간 수, 구간 값, 각 구간에 속한 멀티미디어 객체 수 및 객체의 종류 등으로 특성화 할 수 있다. 또한 채널 시스템의 특성은 채널의 대역폭과 제공 가능한 통과 지연 상한값으로 기술할 수 있다. 여기서는 먼저 두 가지 집합의 시뮬레이션에서 공통적으로 사용한 인자들에 대하여 논의하고, 각각의 시뮬레이션 특성에 맞는 인자는 하부 절에서 설명하기로 한다.

멀티미디어 객체의 종류로는 비디오, 오디오, 이미지, 일반 텍스트 네 가지를 고려한다. 비디오는 MPEG을 이용하여 압축한 후 멀티미디어 서버에 저장되었다고 가정한다. 시뮬레이션을 위하여 MPEG 비디오는 M=3, N=12의 GOP 구조를 갖는다고 가정하고, 각 프레임의 크기를 생성한다[9]. 멀티미디어 문서의 구간 값 지정을 용이하게 하기 위

하여 비디오는 초당 30 프레임을 재생한다고 가정한다. 오디오 객체는 2.184 Kbits의 고정된 크기를 갖는다고 가정하고, 각 오디오 SIU의 재생기간은 비디오 SIU의 재생기간과 동일하다고 가정한다. 텍스트 객체의 크기는 유니폼 분포 U[7.0, 15.0] (Kbits)에서 생성하고, 이미지 객체의 크기는 유니폼 분포 U[91.39, 267.16] (Kbits)에서 생성한다. 채널의 특성을 지정함에 있어 통과 지연은 150-400 ms[10] 사이의 값을 고려한다.

1. 최적해와의 성능비교

최적해와의 성능비교를 위한 시뮬레이션에서는 두 개의 채널이 가용하다는 가정 하에, 한 채널의 대역폭은 64 Kbps로 설정하고, 다른 채널의 대역폭은 4.8 Mbps로 설정한다. 각각의 채널에 대하여 통과 지연을 유니폼 분포, U[0.15, 0.4]에서 불특정하게 산출하면서 총 240개의 멀티미디어 문서 사례에 대하여 시뮬레이션을 수행한다.

시뮬레이션의 결과를 사정하기 위하여 FSA와 BSA의 해의 최적 해에 대한 편차의 오류율, 즉 $(T^{\star}_{max} - T^{*}_{max})/T^{*}_{max}$을 사용한다.

시뮬레이션 결과, FSA의 전체 평균 오류율은 0.093%이었고, 최대 오류율은 1.45%이었다. 한편 BSA의 전체 평균 오류율은 0.106%이었고, 최대 오류율은 2.91%이었다. 그러므로 제안된 두 알고리즘을 사용하면 다양한 멀티미디어 문서 구조 및 채널의 통과 지연에 관계없이 평균적으로 최적해에 근사한 결과를 얻을 수 있음을 알 수 있다.

2. FSA와 BSA 간의 상호 성능비교

제안된 두 알고리즘들 간의 성능 비교는 두가지 측면에서 행한다. 첫 번째 측면은 두 알고리즘의 스케줄링 결과로 나오는 $T^{\star}_{max}$ 값의 비교이고, 두 번째 측면은 생성된 스케줄에 따라 멀티미디어 서버에서 송신한 SIU들이 클라이언트 단에 수신되어 재생될 때까지 버퍼링 되어야 하므로, 클라이언트 단에서 필요한 버퍼 크기의 비교이다.

각 알고리즘의 결과 값인 $T^{\star}_{max}$ 값을 비교하기 위하여 T_{max} 비율을 아래와 같이 정의한다.

$$T_{max}\ 비율 = \frac{T^{\star}_{max} - \delta^{\star}}{멀티미디어\ 문서\ 전체\ 재생시간}$$

여기서 $\delta^{\star}$는 $T^{\star}_{max}$값을 낸 채널의 통과 지연 값이다. T_{max} 비율을 이와 같이 정의함으로써 멀티미디어 문서의 재생시간과 $T^{\star}_{max}$값간의 상관관계를 유출할 수 있다.

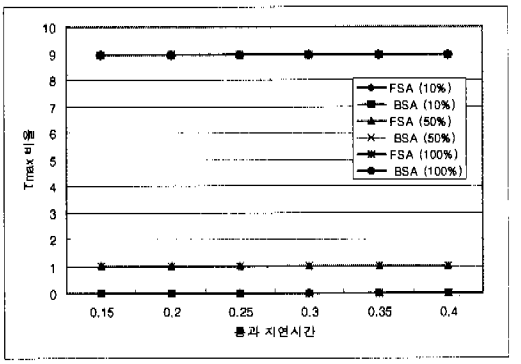


그림 3. 통과 지연이 제안된 알고리즘의 성능에 미치는 영향

먼저 각 채널의 통과 지연이 두 휴리스틱 알고리즘에 미치는 영향을 알아본다. 이를 위하여 연결지향적 망에서 두 개의 채널이 가용하다고 가정한 후, 한 채널의 통과 지연은 150 ms로 고정을 하고 다른 채널의 통과 지연 값을 150-400 ms로 50 ms씩 증가를 하며 변화를 준다. 그 결과가 그림 3에 예시되어 있다. 이 그림에서 FSA(10%)는 멀티미디어 문서를 전송하는 데 필요한 채널용량의 10%만이 가용한 경우를 나타낸다. 이러한 채널 용량 비율은 아래와 같은 식으로 구한다.

$$채널\ 용량\ 비율 = \frac{\sum_j c_j \times 멀티미디어\ 문서\ 전체\ 재상시간}{\sum_i s_i}$$

예를 들어 FSA(10%)는 전체 채널의 처리용량이 멀티미디어 문서에 있는 SIU들의 크기의 합의 10%인 경우를 나타낸다. 그림 3에서 알 수 있듯이 통과 지연의 값이 FSA나 BSA의 성능에 영향을 거의 미치지 않는 것으로 나타났다. 그러므로 이후 시뮬레이션 결과를 분석함에 있어 통과 지연은 고려하지 않고 다른 요소들을 기준으로 결과를 분석한다. 단, 시뮬레이션을 행할 때 통과 지연에 대한 변화는 계속하여 적용한다.

다음은 가용한 채널들의 용량 특성 및 채널 수가 제안된 두 알고리즘의 결과로 나오는 $T^{\star}_{max}$에 미치는 영향을 알아보고, 그림 4에서 결과를 정리한다. 그림 4 (a)는 연결지향적 망에서 두개의 채널이 상이한 용량을 갖는 경우, 한 채널의 용량은 오디오 객체의 재생율에 대응하는 9.6 Kbps로 고정하고, 나머지 채널의 용량을 채널 용량 비율에 의거하여

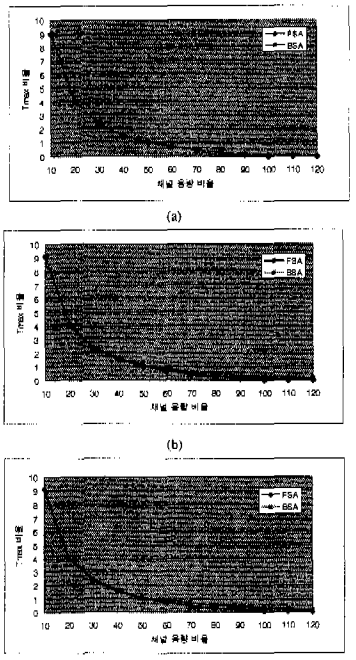


그림 4. 가용한 채널의 용량 특성상 T_{max} 비교 (a) 두 개의 채널이 상이한 용량을 갖는 경우 (c1=9.6 Kbps, c2=채널 용량 비율에 따라 가변), (b) 두 개의 채널이 동일한 용량을 갖는 경우 (c1=c2) (c) 세 개의 채널이 동일한 용량을 갖는 경우 (c1=c2=c3)

변화시켜 그 결과를 본다. 이 경우, FSA와 BSA의 결과가 거의 유사하여, 채널 용량 비율이 100%가 넘는 경우 두 알고리즘의 결과는 동일하고, 채널 용량 비율이 100% 미만인 경우 FSA가 BSA보다 평균 0.3% 정도 좋은 T_{max} 비율을 기록한다. 한편 채널 용량 비율이 100%인 경우 FSA가 BSA보다 15%정도의 좋은 결과를 보인다. 그림 4. (b)에서는 가용한 두개의 채널이 동일한 대역폭을 제공하는 경우와 그림 4. (c)에서는 세개의 채널이 가용하고 그 채널들이 동일한 대역폭을 제공하는 경우에 대한 결과를 보여준다. 이 경우에는 채널 용량 비율이 높아질수록 FSA의 결과가 BSA의 결과보다 좋았으며, 특히 채널 용량 비율이 100%가 넘는 경우에는 BSA의 T_{max} 비율이 FSA의 비율보다 300배 이상 높게 나왔다. 그러므로 T_{max} 비율을 고려하면 SIU 스케줄링 시 FSA를 사용하는 것이 좋은 것으로 결론지을 수 있다.

본 논문에서 고려하는 최적화의 조건이 T_{max}를 최소화하는 데 있지만, 그림 1에서 보여주듯이 망에서 제공하는 자원이 불충분한 경우 T_{max} 만큼의 지연을 멀티미디어 문서 재생 초기에 도입해야 하므로 클라이언트 단에서는 지연을 위한 버퍼가 필요하게 된다. 그래서 그림 4에서 알아본 경우에 대하여 클라이언트 단에서 요구되는 버퍼 크기를 비교하여 본다. 필요한 버퍼 크기도 멀티미디어 문서의 구성에 따라 절대적인 크기가 다르므로, 해당 멀티미디어 문서를 재생하는 데 필요한 버퍼 크기를 전체 멀티미디어 문서를 구성하는 객체의 크기로 평준화시킨 버퍼 수요 비율을 아래와 같이 정의한다.

$$\text{버퍼 수요 비율} = \frac{\text{버퍼 수요}}{\sum_i S_i}$$

버퍼 수요 비율에 관한 결과를 그림 5에 예시한다. 그림 5의 각각의 결과는 그림 4의 분류에 따라 대응하는 결과를 나타낸다. 그림 5에서 크게 두가지 결과를 주목할 만 하다. 우선 FSA와 BSA 모두 채널 용량 비율이 100% 이하인 경우 T_{max}가 감소함에 따라 버퍼 수요 비율도 따라 감소한다는 것이다. 그러나 FSA의 경우 채널 용량 비율이 100%일때 버퍼 수요 비율이 최소로 감소하였다가 채널 용량 비율이 100%가 넘으면 다시 버퍼 수요 비율이 증가한다는

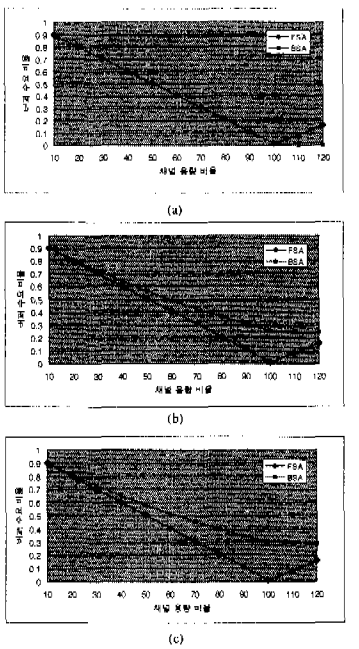


그림 5. 가용한 채널의 용량 특성상 버퍼요구비율 비교 (a) 두 개의 채널이 상이한 용량을 갖는 경우 (c1=9.6 Kbps, c2=채널 용량 비율에 따라 가변), (b) 두 개의 채널이 동일한 용량을 갖는 경우 (c1=c2) (c) 세 개의 채널이 동일한 용량을 갖는 경우 (c1=c2=c3)

것이다. 이러한 현상은 FSA의 알고리즘 단계 FSA 3에서 알 수 있듯이 SIU를 클라이언트 측에 도착하는 시간을 최소화 하는 채널로 스케쥴링을 하기 때문에, 가용 채널 용량이 필요한 용량보다 큰 경우 SIU들을 각각의 재생 마감시간에 맞추어 클라이언트 측에 전송하기보다는 채널의 휴식시간 없이 클라이언트 측에 미리 다 전송해 놓게 된다. 그러므로 필요 이상으로 SIU들이 클라이언트로 먼저 전송되어, 결과적으로 재생 마감시간까지 버퍼링 되어야 한다.

그러므로 위와 같은 결과를 종합하여 보면, 연결지향적 망에서 제공하는 채널들의 대역폭이 상이한 경우, 만일 채널 용량 비율이 멀티미디어 문서를 구성하는 데이터를 전송하는 데 부족하면, 즉 채널 용량 비율이 100% 이하인 경우에는 FSA를 사용하는 것이 효율적이고, 채널 용량 비율이 충분한 경우에는 BSA를 사용하는 것이 효율적이다. 그러나 연결지향적 망에서 제공하는 채널들의 대역폭이 동일한 경우에는 멀티미디어 문서의 전송 스케쥴링에 FSA를 사용하는 것이 효율적이다.

VI. 결 론

본 논문에서, 연결지향적 망이 불충분한 망 자원만을 제공 가능할 때 멀티미디어 동기 유지에 적용할 수 있는 방법에 대하여 고찰하였다. 단 망에서 제공하는 채널들에 대해서는 일정의 QoS를 보장한다고 가정하였다. 이러한 환경에서의 멀티미디어 동기 문제를 해결하기 위하여 멀티미디어 문서의 전송 스케쥴링 문제에 대하여 고찰, 문제를 공식화하였다. 잘 알려진 것처럼 병렬 프로세서의 스케쥴링 문제가 NP-hard 문제이므로, 본 논문에서는 시간적 복잡성이 $O(n \log n + nm)$인 휴리스틱 알고리즘, FSA와 BSA를 제안하였다. 시간적 복잡성에서 n은 멀티미디어 문서를 구성하는 데이터의 개수를 의미하고, m은 연결지향적 망에서 가용한 채널 수를 의미한다.

제안된 두 알고리즘의 성능을 검증하기 위하여 최적해를 구하기 위한 알고리즘을 개발하였으며, 시뮬레이션을 통한 성능 검증 결과 FSA와 BSA 모두 최적해에 근사한 결과를 보임을 알 수 있었다.

또한 본 논문에서는 제안한 휴리스틱 알고리즘들의 상호 성능비교를 다양한 채널 환경에서 알아보았다. 결과적으로 통과 지연 값은 두 휴리스틱 알고리즘의 성능에 영향을 미치지 않음을 확인할 수 있었으며, 망에서 제공하는 채널의 대역폭의 합이 멀티미디어 문서를 전송하는 데 필요한 대역폭보다 적은 경우와, 역 다중화를 사용하는 경우와 같이 각 채널들의 대역폭이 동일한 경우에는 FSA를 선택하여 스케쥴링하는 것이 효율적으로 판단되었다. 하지만 클라이언트 단에서 필요한 버퍼의 크기를 고려할 때, 가용한 채널의 대역폭이 상이하고, 충분한 대역폭이 제공되는 경우에는 BSA를 선택하는 것이 효율적이라는 결과를 얻었다.

참 고 문 헌

[1] G. Blakowski and R. Steinmetz, "A Media Synchronization Survey: Reference Model, Specification, and Case Studies," *IEEE J. Sel. Area. Comm.*, vol 14, no. 1, pp 5-35, Jan. 1996.

[2] K. Ravindran and V. Bansal, "Delay Compensation Protocols for Synchronization of Multimedia Data Stream," *IEEE Trans. Knowl. Data Eng.*, vol. 5, pp. 574-589, Aug. 1993.

[3] Y. Ishibashi and S. Tasaka, "A Group Synchronization Mechanism for Live Media in Multicast Communications," *Proc. of IEEE GLOBECOM 97*, vol. 2, pp. 746-752, Phoenix, Arizona, USA, Nov. 1997.

[4] S. Ramanathan and P.V. Rangan, "Adaptive Feedback Techniques for Synchronized Multimedia Retrieval over Integrated Networks," *IEEE/ACM Trans. Networking*, vol. 1, pp. 246-260, April 1993.

[5] T.D.C. Little and A. Ghafoor, "Multimedia Synchronization Protocols for Broadband integrated Services," *IEEE J. Sel. Area. Comm.*, vol. 9, pp. 1368-1382, Dec. 1991.

[6] M. Woo and A. Ghafoor, "Multichannel Scheduling for Communication of Pre-orchestrated Multimedia Information (Homogeneous Channels Case)," *Proc. of IEEE INFOCOM '94*, vol. 2, pp. 920-927, Toronto, Ontario, Canada, June 1994.

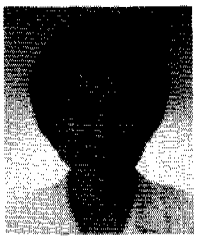
[7] J. Blazewicz, K. Ecker, G. Schmidt, and J. Weglarz, *Scheduling in Computer and Manufacturing Systems*, Springer-Verlag, Berlin · Heidelberg, 1993.

[8] S. Sahni and Y. Cho, "Scheduling Independent Tasks with Due Times on a Uniform Processor Systems," *Journal of the ACM*, vol 27. no. 3, pp. 550-563, Jul. 1980.

[9] A.M. Dawood and M. Ghanbari, "Content-Based MPEG Video Traffic Modeling," *IEEE Trans. Multimedia*, vol. 1, no. 1, pp. 77-87, Mar. 1999.

[10] G. Karlsson, "Asynchronous Transfer of Video," *IEEE Commun. Mag.*, vol. 24, no. 8, pp. 118-126, Aug. 1996.

우 미 애(Miae Woo) 정회원



1985년 2월 : 연세대학교 전자공학과 학사

1991년 12월 : Purdue University, School of Electrical Engineering, 공학석사

1995년 12월 : Purdue University, School of Electrical and Computer Engineering, 공학박사

1985년 1월~1989년 7월: 데이콤 연구원

1996년 1월~1998년 2월: 삼성전자 수석연구원

1998년 3월~현재 : 세종대학교 정보통신공학과 조교수

<주관심 분야> 멀티미디어 통신, 데이터 네트워크, 이동 통신망, 광대역 통신